

События

Урок 2

Рассматриваем события в JavaScript








План курса





Что будет на уроке сегодня

-  Понятие событий
-  Обработчики событий
-  Всплытие и погружение событий
-  Генерация пользовательских событий
-  События клавиатуры и мыши



Обработчик
события





События

Событие — это сигнал от браузера или другой среды исполнения JavaScript, о том, что интересующее нас действие произошло.



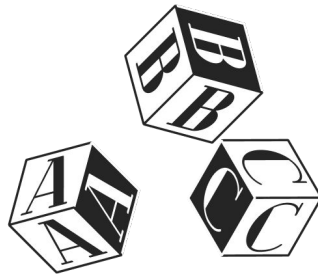
События могут происходить в скриптах



Все элементы DOM-дерева могут производить события



События можно генерировать с помощью специальных методов





Обработчик по атрибуту

```
1 <button onclick="handler()">
```



Используется атрибут HTML-элемента, управление передаётся коду в кавычках. В данном случае – это функция handler



Обработчик по свойству DOM

Обработчик вызывается в коде JavaScript:

- находим элемент DOM (задаем кнопку путем поиска по ID “button”)
- к нему через свойство прикрепляем DOM-обработчик (задаем действие по клику на эту кнопку)

```
1 let button = document.getElementById('button');  
2  
3 button.onclick = () => { } // Обработчик
```



addEventListener

```
1 button.addEventListener(event, handler, options);
```

Специальный метод элемента, который позволяет повесить слушатель на любое доступное событие. С помощью options можно добавить события, недоступные другими способами



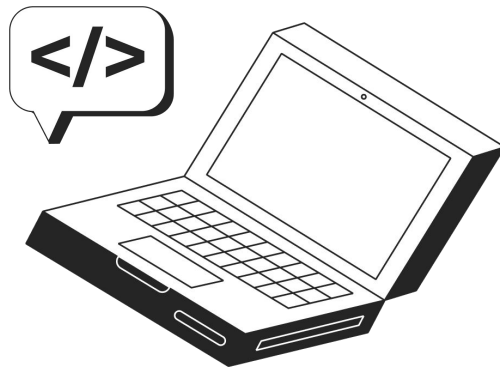
Всплытие, погружение и делегирование событий





Всплытие событий

- 💡 Событие регистрируется в элементе, в котором оно произошло, после – в родителе, и так далее, по цепочке до самого верха
- 💡 Почти все события всплывают
- 💡 Всплытие можно отменить с помощью метода `stopPropagation()`





Погружение событий



Перед тем, как “всплывать”, событие сначала погружается сверху вниз



Погружение происходит автоматически, мы почти не можем им управлять



Поймать погружение событий мы можем только с помощью
`element.addEventListener()`



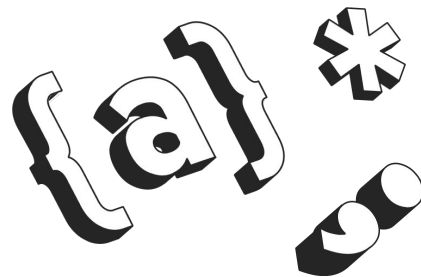
На практике обработка событий на этапе погружения используется редко, но может быть полезна и знать о ней нужно!



Делегирование событий

Делегирование – это приём в разработке, который оптимизирует процесс обработки похожих событий.

Вместо того, чтобы обрабатывать похожие события во множестве элементов, мы вешаем обработчик на общего предка. Благодаря всплытию, он будет выполняться каждый раз, когда это событие будет происходить на любом из его дочерних элементов, а `event.target` позволит обратиться непосредственно к элементу, на котором это событие произошло.





Генерация событий пользователя





Генерация событий пользователя

- ▶ Через конструктор класса `Event` мы можем создать событие.
- ▶ Для запуска события у элемента есть метод `dispatchEvent`.
- ▶ Свойство `isTrusted` нам вернёт ответ, является ли событие встроенным.
- ▶ Свойство `preventDefault` отменит действия браузера по умолчанию.



Обзор событий





События мыши

- 🚩 **mousedown** - кнопка мыши нажата над элементом
- 🚩 **mouseup** - кнопка мыши отпущена над элементом
- 🚩 **click** - кнопка мыши нажата и отпущена над элементом
- 🚩 **dblclick** - двойной клик на элементе
- 🚩 **mouseover** - курсор мыши наведён на элемент
- 🚩 **mouseout** - курсор мыши покинул зону элемента
- 🚩 **mousemove** - произошло любое движение мышью
- 🚩 **contextmenu** - произошёл вызов контекстного меню

Не всплывающие события, которые не разделяют элемент и его потомков.
Считается, что это один элемент:

- 🚩 `mouseenter` // курсор мыши наведён на элемент
- 🚩 `mouseleave` // курсор мыши покинул зону элемента



События клавиатуры

У клавиатуры имеются два основных события:

- `keydown` (при нажатии кнопки)
- `keyup` (при отпускании кнопки)

У события есть свойства с кодом (`code`) нажатой клавиши и её ключом (`key`).

Ключ – это символ клавиши. **Код** – это «физический код клавиши».

Есть отдельные события для нажатых системных клавиш:

```
1 shiftKey // Shift
2 altKey // Alt (или Opt для Mac)
3 ctrlKey // Ctrl
4 metaKey // Cmd для Mac
```



Прокрутка (scroll)

Событие scroll может быть полезно при создании страниц с «бесконечным» содержимым. Например, когда подгрузка следующей страницы происходит при достижении самого низа текущей страницы и встраиванием содержимого ниже. Либо при других потребностях показать дополнительное содержимое.

```
1 window.addEventListener('scroll', () => {  
2     document.getElementById('showScroll').innerHTML  
   = scrollY + 'px';  
3 });
```



Подведем итоги

- 📌 Поговорили об обработке событий
- 📌 Рассмотрели обработку событий на этапах всплытия и погружения
- 📌 Рассмотрели основные методы работы с событиями (делегирование и генерация)
- 📌 Рассмотрели основные интерфейсные события





Спасибо за внимание

